Creative Problem Solving: A Study of Character Rigging in Maya

An Honors Thesis (ART 490)

by

Mollie Connor

Thesis Advisor

Andy Beane

Ball State University
Muncie, Indiana

April 2014

Expected Date of Graduation

May 2014

Abstract

The goal of my thesis is to create a short video, called a demo reel, showcasing my skills as a technical artist. In the animation industry, job applicants are required to submit a demo reel along with their resume to show the potential employer what they are capable of. My demo reel contains my best character rigs and MEL scripts. My reel will give potential employers a demonstration of my skills so they can compare me to other applicants.

Acknowledgements

I'd like to thank Andy Beane for his instruction and encouragement throughout my college career. He has been an incredible resource for knowledge about the animation process and industry. I would also like to thank John Ludwick for challenging me to do my best work.

Artist Statement

I want to work in animation because of the collaborative nature of the industry, as well as the opportunity for specialization. Projects are created via a production pipeline, similar to an assembly line. Within the pipeline, the design team develops the look of the project. The modeling and texturing team build the characters, sets, and props in digital space. Riggers add control systems to characters, and animators create the performance, bringing the characters to life. Breaking down the project into parts means that people must specialize in certain skills to fit into the pipeline. This allows people to find what jobs suit them and gives them a chance to attain mastery in a certain area.

Initially, I loved animation because of the endless possibilities. In 2D animation, the character and environment design is often highly stylized animation, like *The Powerpuff Girls* and *Samurai Jack*. *Avatar: The Last Airbender* is a bit of an exception to the stylization popular in 2D animation. The TV series has extremely detailed background paintings, and although it contains many fantastic creatures, the human characters have realistic body proportions and anatomy. In computer animation, there's a trend towards a more realistic look, like in the *Toy Story* films. At the extreme end of photorealism, visual effects artists can seamlessly add animated characters blended into live-action films, like Gollum in the *Lord of the Rings* trilogy.

Once I started my education in animation, I quickly learned that design and character animation are not for me. The constant critiquing and tweaking, and sometimes even starting over, frustrated me. In character design class, we were all given the same prompt, but we ended up with characters that looked vastly different

from one another. Every design had its own strengths and weaknesses, but there was no universal "right" or "wrong" answer to the prompt.

Rigging is much more suited to my way of thinking. The rigger is given a character that must move in a specific way and needs to create a control system that allows for that type of movement. There is a right answer in this case: either the character moves the way it's supposed to, or it doesn't. Sometimes there is more than one way to get to the right answer. If there are multiple methods of getting to the same solution, I take two things into account to find the best method. First, what is the simplest and fastest solution? Second, what solution will be the most user-friendly for animators? This is where collaboration comes into play. When I build a rig, and then I always try to give it to an animator to test out before calling it finished. Sometimes the animator will find a way to break the rig. When this happens, I'm faced with a new puzzle. Is it possible to fix the rig using existing parts, or is it better to start from scratch? Whatever path I take to fixing it, there is no greater feeling of accomplishment than giving the animator a rig and hearing, "That's exactly how I wanted it to work."

Influences

My rigging methods are strongly influenced by the tutors on the website Digital-Tutors. My favorite tutor is Delano Althais. Delano is responsible for most of the rigging tutorials on Digital-Tutors, so a lot what I know about rigging is what I've learned from him. One of my favorite things about Delano is his enthusiasm. He honestly thinks rigging is one of the coolest things in animation. In rigging, it's good practice to test a control before starting to build another one. When Delano is testing controls, he often

will say something like, "Look at that. Isn't it great?" or "Wow! That's working just beautifully." Completing a full rig is a big task, but taking it in parts and celebrating the small victories makes it manageable and enjoyable.

Another thing that I've learned from Delano is the importance of knowing how and why something works the way it does. For example, most of the time I connect my controllers to their joints using constraints, but when I'm building a foot control using the reverse-lock method, I want to parent the foot joint to the controller instead of a constraint. The reason that I do this is to free up the transformation channels on the foot joints so they can be connected to other things. This gives the animators another level of control over the movement of the foot. By understanding the way the reverse-lock foot works instead of just memorizing that it requires a parent instead of a constraint, I'm able to apply this information to other problems.

Description of Work

My demo reel contains three rigging projects from my senior year: the android used in our class's short film, a running shoe that I put together for a product visualization project, and a cartoon-style dog with squash and stretch functionality. I also have included two scripts. The first script opens up a window that allows the texture artist to adjust the attributes on multiple shaders at once. The second script automates part of the process I used to create the facial controls on the android rig.

Android Rig

The android rig sounded like a simple project at first. Because the character was mechanical instead of organic, there was no skinning involved. Skinning is a way to attach a character to the joint system that allows for smooth, realistic deformations. It also is time-consuming and involves a lot of testing and tweaking. One of the challenges of creating the android rig was the number of parts involved. Most models I had worked with in the past had one piece of geometry for the body, one piece for each article of clothing, and occasionally a separate piece of geometry for the head. The android rig had over a hundred separate pieces of geometry. Instead of trying to keep track of the connections on every piece, I used a system of groups and parents to simplify the rig. I created a group under each bone in the skeleton and added geometry to the groups based on which bone they were supposed to follow. This method had the added bonus of allowing the modeler to add, remove, or change parts without breaking connections between the geometry and the rig.

Another challenge of the android rig was the pistons in the neck, torso, and shoulders. These pistons weren't directly controlled by a single bone. Rather, their position was affected by movement of multiple bones, with the added restriction that the two halves of the piston had to stay in line with each other. The solution of this problem was surprisingly simple. For each half of the piston, I created a locator within a group. The group was parented to the bone it was supposed to follow. The locator was aim-constrained to the locator on the other half of the piston. The piston geometry was parent-constrained to the locator. This means that when the arm moves away from the chest, the piston in the shoulder follows and automatically reorients itself.

The face of the android was different than the rest of the body because it was meant to deform like an actual human face. An easy way to rig a face is to use blend shapes, which are saved poses that you can turn on and off as well as combine with other poses. This method limits the animator because they can only use the pre-made poses. Another option is to use a joint-based facial rig, which involves placing a skeleton within the face and giving the animator separate controls for the brows, eyes, nose, mouth, etc. This method gives the animator more freedom, but it is also more time-consuming for the animator. For the android's face, I created a rig that had blend shape and individual controller capabilities. I created curves for the brows, eyelids, cheeks, and lips. Then I created locators that were attached to the curves using a motion path, which means that when the curve was moved or deformed, the locators would follow. Next I created a joint for each locator and parented the joint beneath its locator. Each joint was duplicated, and the duplicate was parented below the original. The face was skinned to the duplicate (offset) joints. I created blend shapes for the face using the curves and added slider controls so the animator could access those poses. Finally, each offset joint received its own controller, so if the animator wanted to adjust a pose or create their own pose, they had the option to do that.

Face Curve Control Builder

The process of creating the face rig took a long time. There were about a dozen curves, and each curve drove the motion of either three or five locators. Each locator had a double joint parented underneath it, and then a controller was created for each offset joint. Plus most of the parts had to be renamed. The steps were simple but repetitive, so I decided to write a script to automate most of the process, simplifying it

into three easy steps through a UI window. The window is broken up into different sections for each step. The first step is the creation of the joint-control hierarchy. The rigger chooses the size of the locators and control objects, and then hits "CREATE" to build the hierarchy. The option for changing the size of locators and controllers is important because if the model to be rigged is just a floating head, the face is probably going to be larger than if the head is attached to a full character. It follows that the floating head model requires larger controllers than the full character model. By allowing the rigger to choose the size of the control during creation, it saves them time because they don't have to go back and resize every controller if the default size doesn't suit the rig. The second step is to attach the locators to the curve. The rigger selects their curve, followed by the locator they want to attach. In the UI window, they can use a slider bar to choose a value between 0 and 2 that corresponds to points along the curve. Hitting "ATTACH" connects the locator to the curve at the desired value and zeroes out any unwanted rotation values the locator may pick up during the connection process. The final step of the process is renaming the objects. The rigger enters their desired suffix into a text field in the UI window and hits "RENAME" to add the suffix to the most recently created joint-control hierarchy. I added an extra function to the UI window to make the skinning process a little easier. In the typical skeleton for a rig, all or most of the joints are parented under a root joint. When it's time to skin, I can just select the root joint and type in a short command to select every joint within that hierarchy. With the joint-based facial rig, each joint-control hierarchy is a separate entity, so each joint must be selected individually. The way my script works, the name of each bind joint starts the same way, so I was able to use Maya's wildcard function so

search for and select every joint with the prefix "jntControl." This allows the rigger to select all bind joints with just one click.

Squash and Stretch Dog Rig

Some of the most fun animation is highly stylized and cartoony. For example, in *Tom and Jerry* shorts, the characters are often stretched out or squashed into comical poses. In 2D animation, the animators can just draw the characters off-model to get those poses, but it requires some extra work on the rigging level to get that type of functionality in 3D animation. To practice this skill, I created a simple wiener dog model that would have stretch functionality on its torso and legs. On the legs, I used an IK setup, which means that the animators move around the foot, and the thigh and knee bend automatically. I created a node that measures the distance between the thigh bone and the foot bone. I stored the distance when the leg is fully extended in a variable called the stretch threshold. Then I created an expression that would rescale the leg bones in according to the relationship between the stretch threshold and the current distance between the thigh and foot. If the distance was greater than the stretch threshold, the leg gets longer, but if the distance is less than or equal to, the leg bones keep their original scale to allow for the leg to bend.

When I was testing the rig, I ran into a problem. When increasing the size of the character using the world controller, the legs began to stretch. This is because the increase in size made the distance node value larger than the stretch threshold, activating the stretch expression. To fix this, I attached a division node to the stretch threshold. With the help of another expression, I was able to make the division node

calculate the new stretch threshold relative to the scale of the world controller, which eliminated the unexpected stretching.

The Cecily Script

The second script in my demo reel is "The Cecily Script." The idea for this script came about while working on the senior class short film. Our texture artist, Cecily, needed to adjust some of the settings on the android's textures so they would render properly. There were about 40 different textures on the android, and they all needed to be changed uniformly. I asked Cecily for a list of attributes and the values they needed to be changed to and wrote up a short script that would find every Blinn texture in the scene and change their attributes according to Cecily's list. Later on, Cecily needed to change the attributes again, so I altered a couple lines of the script to fit her new list. My script wasn't very user-friendly for people who aren't familiar with scripting. I created a new script that opens a UI window that contains a list of attributes with slider controls. The texture artist chooses their values using the sliders, and then they have the option of applying that value to every Blinn in the scene, or they can choose which Blinns they want to change on a selection basis. This script is important because it shows that I can help speed up other people's workflow within a production pipeline.

Conclusion

I chose my projects for my demo reel to show a variety of skills. The android rig is first in my demo reel because it is my best work. The automated piston movement and the hybrid blend shape/joint facial rig make it more than just a standard biped rig. The dog rig shows that I am comfortable rigging non-human characters and that I

understand how to use expressions to add functionality to my rigs. The Cecily Script and the facial curves script show an understanding of MEL scripting. It also shows that I can be useful in creating tools to save time. My ability to do many types of projects will help me prove to a potential employer that I can fit in to their production pipeline.

Reel Breakdown

1. Android rig | 0:03 – 1:23

   Model made by Andrew Kleine for *Vestige* student film. Rigged by me.

2. Face Curve Control Builder script | 1:24 – 2:08

   Script written by me in MEL.

3. Squash and Stretch Dog rig | 2:09 – 2:52

   Animated by Cale Royer. Modeled and rigged by me.

4. Nike Free shoe rig | 2:53 – 3:17

   Modeled, textured, rigged, and animated by me.

5. The Cecily Script | 3:18 – 3:35

   Script written by me in MEL.